Optimización: Trabajo Práctico Final

Matías Benedetto - Valeria Di Tomaso

Primer Cuatrimestre 2017

1. Minimización sin restricciones: Problema del Polígono de Área Máxima

El objetivo del problema consiste en encontrar el polígono de n lados (con diámetro menor o igual a 1), con el área máxima. A continuación se plantearán dos modelos distintos, resueltos mediante el método del gradiente y el algoritmo de Broyden–Fletcher–Goldfarb–Shanno, y se compararán los resultados obtenidos. Si bien ambos modelos sí tienen restricciones, se resolvieron como si no las tuvieran, utilizando el método de penalización.

1.1. Modelo 1

El presente modelo fue obtenido de 'Benchmarking Optimization Software with COPS 3.0'[1].

Maximizar $\frac{1}{2}\sum_{i=1}^{n-1} r_{i+1}r_i sen(\theta_{i+1}-\theta_i)$

sujeto a:

1)$r_i^2 + r_j^2 - 2r_i r_j cos(\theta_i - \theta_j)$ con $1 \leq i \leq n, 1 \leq j \leq n$

2) $\theta_i \leq \theta_{i+1}$ con $1 \leq i < n$

3) $\theta_i \in [0, \pi]$ con $1 \leq i \leq n$

4) $r_i \geq 0$ con $1 \leq i \leq n$

5) $r_n = 0$

6) $\theta_n = \pi$

7) $r_i \leq 1$

En este modelo se asume que el último vértice coincide con el origen de coordenadas. Es por este motivo que se fijan las últimas tres condiciones.

Por su parte, la función objetivo surge de la fórmula de área de un polígono obtenida a través del método de determinante de Gauss. Es decir, se divide el polígono en triángulos, numerando los vértices en sentido antihorario, y se calcula el área de cada uno de ellos mediante el determinante. De este modo se llega a la siguiente fórmula:

$$\text{Área}=\frac{1}{2}\left(\begin{vmatrix} x_1 & x_2 \\ y_1 & y_2 \end{vmatrix} + \begin{vmatrix} x_2 & x_3 \\ y_2 & y_3 \end{vmatrix} + \ldots \begin{vmatrix} x_n & x_1 \\ y_n & y_1 \end{vmatrix}\right)$$

En este caso estamos tomando el último vértice en el origen, por lo que el último término desaparece. Además, trabajamos con coordenadas polares, de donde se deduce que:

$$\begin{vmatrix} x_i & x_{i+1} \\ y_i & y_{i+1} \end{vmatrix} = r_{i+1}r_i sen(\theta_{i+1}-\theta_i)$$

Por otro lado, la restricción 1) impone que el diámetro sea menor que 1, y surge de tomar la norma entre dos puntos cualquiera (expresados en coordenadas polares); mientras que la restricción 2) obliga a que los vértices estén ordenados en sentido antihorario.

[1] http://www.mcs.anl.gov/ more/cops/cops3.pdf

1.2. Modelo 2

El modelo anterior presenta una gran cantidad de mínimos locales, del orden de $n!$, por lo que con algunos métodos de minimización es difícil alcanzar el óptimo global.

La alternativa que propondremos a continuación supone que todos los vértices están circunscriptos en una circunferencia de radio $\frac{1}{2}$. De esta manera, al eliminar todos los r_i, estamos reduciendo la cantidad de variables totales a la mitad. Naturalmente, como estamos maximizando el área con vértices circunscriptos, podría darse que el diámetro sea estrictamente menor que 1. En ese caso realizaremos un reescalamiento para que sí alcance dicho valor.

En principio, la hipótesis es que el modelo es heurístico, porque a la hora de resolver el problema podría ocurrir que el máximo obtenido pueda reescalarse en una proporción menor que una solución que tiene menor área inicial pero al reescalarse, supera en área a la primera. Es decir, al separarse el problema en dos instancias: maximización y reescalamiento, puede llegar a perderse optimalidad. De todas formas, al iterarse muchas veces el procedimiento de hallar máximos, se puede evadir esta dificultad.

A continuación detallaremos el modelo:

Maximizar $\frac{1}{8} \sum_{i=1}^{n-1} sen(\theta_{i+1} - \theta_i) + sen(\theta_1 - \theta_n)$

sujeto a:

$1')$ $\theta_i \leq \theta_{i+1}$ con $1 \leq i < n$

$2')$ $\theta_i \in [0, 2\pi]$ con $1 \leq i \leq n$

$3')$ $\theta_n = 2\pi$

La función objetivo se obtiene con el mismo razonamiento que en el modelo anterior, más el hecho de que en este caso $r_i = \frac{1}{2}$ para todo $1 \leq i \leq n$. Dado que ahora los vértices estarán en la circunferencia de radio $\frac{1}{2}$, la restricción 1) del modelo anterior puede obviarse. Lo mismo ocurre, naturalmente, con las restricciones que involucraban r, es decir los puntos 4), 5) y 7). Otra modificación realizada al modelo anterior es que ahora el último vértice no estará en el origen de coordenadas, y por lo tanto ahora tiene más sentido tomar la restricción $3')$

Como puede observarse este modelo resulta más simple, y en la siguiente sección veremos que no por ello pierde eficiencia.

1.3. Resultados

En la Tabla 1 podemos observar los resultados obtenidos para el primer y segundo modelo con el algoritmo de BFGS, mientras que en la Tabla 2 exponemos los resultados esta vez calculados con el método del gradiente.

En ambas tablas, en la columna Reescalamiento se indica en qué proporción pudo reescalarse la solución obtenida con el modelo 2, en donde los vértices estaban circunscriptos.

Todos los resultados fueron obtenidos corriendo el algoritmo 100 veces, y conservando la mejor solución. De esta manera se pueden filtrar los posibles óptimos locales que hallen los métodos. Para la búsqueda del paso de descenso, se utilizó la función fminsearch provista por Matlab. Mediante un simple análisis, se comprobó que el criterio de parada de la misma fue el hecho de haber encontrado un mínimo local.

Los gráficos con los polígonos obtenidos pueden observarse en las figuras 1, 2 y 3.

Notar que dado que el diámetro es menor o igual que 1, el área del polígono tiene como cota superior el área de la circunferencia de radio $\frac{1}{2}$, es decir $\frac{\pi}{4}$.

Está probado que para n impar y $n = 4$, el polígono que maximiza el área es regular. Esto evidencia que para triángulos los cuatro algoritmos arrojan resultados satisfactorios, muy próximos al área del triángulo equilátero unitario, que es de $\frac{\sqrt{3}}{4} \approx 0,433$. Para poder comparar los resultados con el óptimo global, se realizó una función que calcula el área de polígonos regulares con cantidad de lados impar y de diámetro 1.

El método del gradiente obtuvo resultados levemente mejores, aunque el algoritmo demoró más en encontrar una solución. En el caso del cuadrilátero, en donde el área teórica es $\frac{1}{2}$, vemos que el modelo 2

n	Área óptima	Área Mod. 1	Pen. Mod. 1	Área Mod. 2	Pen. Mod. 2	Reescalamiento
3	0.433	0.4328	2.61 e-11	0.43293	1.27 e-15	1.1546
4	0.5	0.4999	1.37 e-11	0.5	4.83 e-16	1
5	0.6572	0.6291	9.55 e-13	0.65279	3.49 e-14	1.049
6	-	0.6564	1.99 e-11	0.64951	6.13 e-13	1
7	0.7197	0.6778	2.62 e-11	0.7073	6.45 e-13	1.017
8	-	0.6844	1.15 e-11	0.7069	2.59 e-13	1

Cuadro 1: Comparación de resultados entre el Modelo 1 y 2 para el algoritmo de BFGS

n	Área óptima	Área Mod. 1	Pen. Mod. 1	Área Mod. 2	Pen. Mod. 2	Reescalamiento
3	0.433	0.4330	2.14 e-11	0.4331	2.63 e-17	1.1547
4	0.5	0.4999	2.02 e-11	0.5	3.94 e-17	1
5	0.6572	0.6571	2.01 e-11	0.65716	6.45 e-17	1.0515
6	-	0.6749	2.01 e-11	0.64952	3.97 e-16	1
7	0.7197	0.7196	2.05 e-11	0.71974	5.50 e-16	1.0257
8	-	0.7267	2.01 e-11	0.70711	4.49 e-15	1

Cuadro 2: Comparación de resultados para el Modelo 1 y 2 para el método del gradiente

devolvió un cuadrado, mientras que el modelo 1, si bien aproximó bastante bien al máximo global, obtuvo un cuadrilátero irregular.

Para el caso $n = 5$, la mejor solución fue obtenida por el modelo 2 resuelto mediante el método del gradiente. La única instancia que tuvo una performance más pobre fue la del modelo 1 con BFGS. Un comportamiento similar puede observarse para el caso $n = 7$.

Otro hecho destacable es que el modelo 2 tiene a dar polinomios más regulares, incluso para $n = 6$, que es par. Acá es bastante claro que el modelo 1 arroja soluciones más satisfactorias tanto para el método del gradiente como para el de BFGS. Se corrió el modelo 2 varias veces para $n = 6$, y aproximadamente el 50 % devolvió polígonos regulares.

En términos generales, el método del gradiente obtuvo soluciones mejores que el de BFGS. Sin embargo, este insumió menos tiempo. Para $n's$ chicos, la diferencia es imperceptible, pero a medida que aumenta, la brecha naturalmente va aumentando.

Con respecto al modelo utilizado, el 2 pareciera tener mayor eficiencia, sobre todo para $n's$ impares.

También se corrió el modelo para $n = 32$, pero no se lo incluyó en la tabla puesto que al tener más variables los tiempos de corridas eran demasiado extensos, y sólo se realizaron 10 iteraciones. El modelo 1 en este caso arrojó soluciones muy pobres, con polígonos no convexos. Esto era completamente esperable dado que teóricamente hay cerca de 32! mínimos locales. En este caso el modelo 2 sí obtuvo una solución convexa bastante razonable incluso en tan poca cantidad de iteraciones.

2. Minimización con restricciones: Problema de la trayectoria de una pelota de golf

2.1. Descripción del modelo

En este problema se busca optimizar la trayectoria de una pelota de golf en un campo no plano, de manera tal que se minimice la velocidad tangencial final con la que la pelota arriba al hoyo. Se lo estudia como un problema de minimización no convexo, como contrapartida del enfoque basado en dar un vector de velocidad inicial y resolver por integración de las ecuaciones de movimiento para determinar si la pelota llega al hoyo y con qué velocidad. Este último enfoque tendría sólo dos variables a optimizar, pero es más complicado de resolver. Es por eso que se eligió otra estrategia, que a pesar de un mayor tamaño en la

cantidad de variables, resulta de dificultad comparable, y cuenta con la ventaja de tener más control sobre lo que sucede al tener las variables y la física expresadas en los mismos términos. Se sigue lo detallado en [2] como referencia. Se discretiza el tiempo total en N subintervalos de tiempo y se definen las siguientes variables del problema para $i = 0, ..., N$:

- T: tiempo total de la trayectoria
- x(i), y(i), z(i): posiciones a tiempo i
- $v_x(i)$, $v_y(i)$, $v_z(i)$: velocidades a tiempo i
- $a_x(i)$, $a_y(i)$, $a_z(i)$: aceleraciones a tiempo i
- $Fr_x(i)$, $Fr_y(i)$, $Fr_z(i)$: Fuerzas sobre la pelota a tiempo i
- $N_x(i)$, $N_y(i)$, $N_z(i)$: componentes del vector normal a tiempo i
- Nmag(i): magnitud de la fuerza normal a tiempo i
- S(i): rapidez (norma del vector velocidad) a tiempo i
- dzdx(i): derivada de la superficie con respecto a x en la posición a tiempo i
- dzdy(i): derivada de la superficie con respecto a y en la posición a tiempo i
- E(i): Energía total a tiempo i

Se utilizaron los siguientes parámetros tomados constantes:

- $m = 0{,}01$ g: masa de la pelota
- $g = 9{,}81$ m/s^2: aceleración de la gravedad
- $\mu = 0{,}07$: coeficiente de fricción.

A través de las restricciones, lineales y no lineales, que vinculan las variables, se introducen diversos conceptos físicos. Se consideró el método trapezoidal, mediante el cual se discretizan las componentes de posición, velocidad y aceleración en los mismos momentos de tiempo y se constringe a que el promedio de dos valores adyacentes sea igual a la diferencia apropiada. Por ejemplo, se impone que $(v_x(i) + v_x(i-1))/2 = (x(i) - x(i-1))/(T/N)$ para $i = 1, ..., N$, y análogamente para las demás componentes y para la aceleración. Se numeran a continuación cualitativemente las restricciones impuestas:

- La pelota debe estar sobre la superficie en todo momento
- La pelota debe permanecer en el campo
- Relación entre velocidades y posiciones dada por las leyes de Newton
- Relación entre aceleraciones y velocidades dada por las leyes de Newton
- Las únicas fuerzas son la gravitatoria, la fuerza normal ejercida por la superficie y las fuerzas de rozamiento.
- La fuerza normal está dada por la proyección de la fuerza gravitatoria y por la componente normal de la aceleración.
- El rozamiento es proporcional a la magnitud de la fuerza normal en dirección de la velocidad y en sentido opuesto.

Para resolver el problema se utilizó la función de Matlab *fmincon* ajustando el número de iteraciones de acuerdo a N. La función objetivo a minimizar es la velocidad en el plano (i.e., $v_x^2 + v_y^2$). Sin embargo, para evitar problemas numéricos es preferible ajustar la función objetivo para que la velocidad final en el plano sea cercana a un valor pequeño.

[2] CASE STUDY IN TRAJECTORY OPTIMIZATION: PUTTING ON AN UNEVEN GREEN - Robert Vanderbrei

2.2. Resultados

En esta sección se presentan resultados para 3 superficies distintas. Se grafican la superficie, sus límites, la trayectoria junto con la velocidad instantánea y los puntos inicial y final.

2.2.1. Problema 1

Para el primer ejemplo se toma el propuesto en el trabajo citado, que consta de dos superficies prácticamente planas unidas por una loma de pendiente moderada. El resultado obtenido con una discretizan en 30 pasos se muestra en la Figura 4. El mismo coincide con lo esperado. El tiempo final del recorrido fue aproximadamente 3,5. Se muestra además un gráfico con valores de velocidad, altura y energía en función del tiempo (Figura 5). Se nota que durante el recorrido existe una pérdida de energía debida a la fricción, y que la velocidad en el plano prácticamente coincide con la rapidez debido a que la superficie cerca del hoyo prácticamente no tiene pendiente.

2.2.2. Problema 2

En este problema se considera una superficie de forma sinusoidal que termina en una región plana. En una de las direcciones no hay pendiente y la bola parte en bajada del punto inicial. Como las coordenadas de los puntos inicial y final en la dirección sin pendiente no son las mismas, se obtuvo una trayectoria que toma la subida de manera inclinada, como era de esperar. Se discretiza la trayectoria en 20 puntos y se muestran los resultados en las Figuras 6 y 7.

2.2.3. Problema 3

El último ejemplo trata de una superficie en forma de parábola sin pendiente en una dirección. El análisis en este caso consiste en mantener fijo el punto final mientras se varía el inicial. De este modo se pueden comparar las trayectorias optimizadas obtenidas (Figuras 8 y 9). Se presentan también en la Tabla 2.2.3 otros datos comparativos de las trayectorias estudiadas.

Punto inicial	Distancia	Tiempo	Rapidez promedio	Rapidez máxima	Energía inicial
A	6.2271	4.3952	1.4168	0.13476	0.070969
B	2.8431	1.6442	1.7291	0.010971	0.044549
C	5.6862	2.5765	2.2069	0.0097466	0.065096
D	2.5154	1.827	1.3768	0.88712	0.049123
E	6.8138	2.2001	3.097	2.1766	0.09722

3. Minimización de funcionales: Problema de Micromagnetización

El micromagnetismo es una rama de la física que se encarga de predecir el comportamiento magnético de los materiales a una escala micróscopica.

La idea de este problema es encontrar la configuración magnética estática o de equilibrio. Definimos el campo magnético como $\mathbf{M}(\mathbf{r})$, con $\mathbf{r}$ vector posición en $\mathbb{R}^3$. Para obtener el estado de equilibrio, se debe minimizar la energía total, que está dada por:

$$E_{tot} = E_{exch} + E_{anis} + E_{mag}$$

donde

$$E_{exch} = \int_V A[(\nabla M_x)^2 + (\nabla M_y)^2 + (\nabla M_z)^2]dV$$

es el término de energía de intercambio, con A constante;

$$E_{anis} = \int_V KV sin^2\theta$$

es el término de energía de anisotropía, con K constante, V el volumen de la muestra y θ el ángulo entre la magnetización y la dirección preferida (easy direction)(en este caso se tomó la energía de anisotropía uniaxial);

$$E_{mag} = -\int_V \left[\mathbf{M} \cdot \left(\mathbf{H_a} + \frac{H_d}{2}\right)\right] dV \tag{1}$$

es el término de energía de magnetización, con $\mathbf{H_a}$ el campo externo aplicado, que fue tomado como constante, y

$$\mathbf{H_d} = -\int_V \frac{(\mathbf{r}-\mathbf{r}')\nabla \cdot \mathbf{M}(\mathbf{r}')}{|\mathbf{r}-\mathbf{r}'|^3} dV' + \int_S \frac{(\mathbf{r}-\mathbf{r}')\ \mathbf{M}(\mathbf{r}')}{|\mathbf{r}-\mathbf{r}'|^3} dS'. \tag{2}$$

Este último término asociado con la magnetización, a diferencia de los anteriores, es global. Para su cálculo por lo tanto se requieren dos pasos. El primero consiste en obtener el valor de $\mathbf{H_d}$ en todos los baricentros de los tetraedros. Para realizar este cálculo se procede a integrar numéricamente la ecuación 2 con r' en cada baricentro. Para la integración volumétrica se usó una regla de Gauss de primer orden (un sólo punto de integración), al igual que con las integrales de flujo sobre la frontera del dominio. Cuando se debe integrar en el elemento al cual pertenece el r' donde se quiere obtener $\mathbf{H_d}$, se utiliza una regla de Gauss de segundo orden (8 puntos en 3D). Con esto se evita evaluar la integral sobre la singularidad. Como la discretización del campo es de primer orden, la velocidad de convergencia del resultado al refinar no se ve afectada por el orden de las integraciones.

Una vez obtenido $\mathbf{H_d}$, se hace una aproximación numérica de (1) calculando el valor de la energía en el baricentro y multiplicándola por el volumen del tetraedro.

Se eligieron estos órdenes de aproximación de integrales por un criterio de velocidad, ya que el tiempo requerido para órdenes más altos, juntos con el hecho de que se realizan para cada paso del algoritmo de minimización, y que el cálculo de $\mathbf{H_d}$ crece como N_{DOF}^2, hacen casi impracticable el uso de integraciones más precisas.

3.1. Estrategia de resolución

Dado que se asume que la magnetización es un campo continuo, un posible modo de abordar el problema consiste en aplicar el método de elementos finitos, usando la base $\{1, x, y, z\}$.

De esta manera, se discretiza el espacio en una malla, y dado un valor de la magnetización sobre los nodos, se interpola el valor en cada uno de los tetraedros a partir de la base antes mencionada.

Para minimizar la energía total se implementaron tres tipos de algoritmos distintos: algoritmos genéticos, de descenso aleatorio, y el método de recocidos simulados.

3.1.1. Algoritmo genético

Se implementaron diversas variantes con diferencias sutiles. Este tipo de algoritmo se basa en el concepto biológico de la evolución. Tenemos una población inicial conformada por individuos de distinta aptitud. A lo largo de las sucesivas generaciones los individuos más aptos se van reproduciendo pasando sus características a sus hijos. Sin embargo de generación a generación también pueden producirse mutaciones.

Consideramos dos tipos de reproducción: por ADN y por combinación convexa. La primera forma es muy similar a la idea biológica de que cada uno de nuestros genes es la herencia de uno de los de nuestros padres. Obviamente en biología esta idea es un poco más compleja pues existe el concepto de genes recesivos y dominantes que hace que algunas características sean más predominantes que otras, por lo que nuestra idea es una mera simplificación. La misma consiste en, dados dos padres elegidos al azar, generar un hijo de la siguiente forma: cada una de las coordenadas del punto se hereda de la madre o del padre con igual probabilidad. En la literatura este método se conoce como Crossover Uniforme.

La segunda forma no se basa tanto en una idea biológica, sino que es más inherente a nuestro problema. Surge un poco de la suposición de que si tenemos dos soluciones buenas un valor entre ambos puede llegar a ser también razonable. De este modo, lo que hacemos es elegir un valor λ al azar entre 0 y 1 y consideramos:

$$X_h = \lambda X_p + (1 - \lambda)X_m$$

donde X_h, X_p, X_m son el hijo, el padre y la madre, respectivamente. Notar que en realidad el hecho de distinguir entre una madre y un padre es un mero formalismo, ya que un mismo individuo puede cumplir ambas funciones.

Otro aspecto de la reproducción es que, si bien de cada pareja surge un único hijo, en principio un individuo podría llegar a reproducirse nuevamente con otro distinto, y en casos muy aislados incluso volver a juntarse con quien lo hizo inicialmente. Esto se debe a que no separamos a los individuos una vez que se reproducen.

Por otro lado, como queremos que la cantidad de individuos en cada iteración se mantenga estable, elegiremos una estrategia elitista de selección: todos los padres pasan a la siguiente generación, a la cual le agregamos a sus hijos (incluyendo a los surgidos a partir de mutación), hasta completar nuevamente la población con el mismo tamaño que la de la generación anterior.

Con respecto al tipo de mutaciones también consideramos dos variantes: azar y gaussiana. El caso al azar consiste en generar un punto equiprobable del dominio, lo cual no tiene mucha relación con ningún individuo de la población, constituyendo una perturbación un tanto extrema. La segunda variante quizás tiene un poco más de analogía con el fenómeno biológico: tomamos un individuo al azar y perturbamos alguna de sus coordenadas con un número aleatorio distribuido como una normal estándar. Esta modificación resulta mucho más sutil que la anterior.

3.1.2. Recocido simulado

Este método es similar al método de descenso en el sentido que se generan puntos al azar y se guarda la mejor solución hasta el momento. Pero ahora existe una diferencia: mientras en el método de descenso nuestra sucesión de soluciones generadas en cada iteración es estrictamente decreciente, ahora vamos a tener la posibilidad de empeorar nuestra solución con una determinada probabilidad, que va bajando a lo largo del tiempo. Esto resulta más efectivo para hallar un mínimo global, porque evita estancarse en uno local.

Dado que el método surgió a partir de la física, vamos a considerar una variable T positiva, denominada temperatura. La misma es utilizada para definir la siguiente función de probabilidad:

$$P(T, X, S) = e^{\frac{f(X)-f(S)}{T}}$$

donde X es un punto generado al azar, y S es la mejor solución hallada hasta el momento.

Notar que si X es una solución que empeora la actual, entonces el exponente es negativo. Luego si el denominador T es suficientemente grande con respecto al numerador entonces el exponente es cercano a 0. De este modo, la función es cercana a 1. Como queríamos que la probabilidad de aceptar una peor solución fuera decreciendo a lo largo del tiempo, entonces vamos a ir achicando T de manera tal que el exponente sea más negativo y por ende disminuya el valor de la función de probabilidad. Sin embargo hay que tener especial cuidado a la hora de hacerlo, pues se corre el riesgo de desaprovechar la ventaja del método. Lo más recomendable es ir decreciendo muy lentamente, es decir sólo dada una cantidad de iteraciones, y de manera muy sutil.

3.1.3. Descenso aleatorio

Se programaron tres variantes de este método: descenso más rápido, con búsqueda local y búsqueda local iterada.

El primer método consiste en, dada una función con su respectivo dominio, generar números al azar e ir guardando aquel en el cual la función tiene un valor menor. Naturalmente, la cantidad de puntos

generados para obtener una solución razonable dependerá mucho del dominio donde estemos trabajando. Por ejemplo, si consideramos que la función está definida sobre el cuadrado unitario de $\mathbb{R}^2$, como asumimos que los puntos generados al azar siguen una distribución uniforme, es razonable por ejemplo, que si iteramos unas 100 veces, obtengamos una solución con un error esperado de 1/10. En cambio, si trabajamos con dominios más grandes, 1000 iteraciones pueden no darnos una solución buena.

El segundo método con búsqueda local consiste en ir generando números al azar, pero ya no distribuidos uniformemente sobre todo el dominio de la función, sino en un entorno de la mejor solución obtenida hasta el momento. Para generar estos vecinos definimos ε de manera tal que sea una proporción pequeña de la longitud del intervalo de definición.

El último de los métodos, el de búsqueda local iterada, es una combinación de los dos anteriores. La idea subyacente es tomar la mejor solución obtenida hasta el momento y moverse localmente una cantidad determinada de veces, como se hacía en el segundo método. Sin embargo, ahora, cada tanto se genera un punto al azar en cualquier parte del dominio y se vuelve a realizar la búsqueda local sobre el mismo. Esta generación de puntos también se repite una cantidad determinada de veces, realizándose así dos ciclos anidados, motivo por el cuál es más costoso computacionalmente.

3.2. Resultados

En todos los casos se trabajó con una malla con paso de discretización $h = 0{,}25$, y los siguientes parámetros: $A = 20$ como constante del término de intercambio, $K = 100$ como constante de anisotropía, $v = (0, 1, -3)$ como easy direction, y $H_{ap} = (1, 1, 1)$.

En el gráfico 10 se encuentra la configuración de magnetización que se tomó como punto inicial para algunos de los algoritmos. La misma tiene una energía total de 50.42. En el análisis realizado sólo tres de los algoritmos pudieron mejorar ese valor. Cabe aclarar que dos de ellos lo hicieron porque la tomaron como punto inicial.

Para el algoritmo genético, dada la dimensión del problema, calculamos sólo 50 generaciones compuestas por una población de 40 individuos. En promedio cada corrida insumió 2 hs. Aumentar estos parámetros hubiera dado resultados quizás más satisfactorios, pero hubiera sido muy costoso a nivel computacional.

Para el caso del algoritmo de descenso aleatorio más rápido y descenso local se realizaron 100 iteraciones, y en particular en el segundo se tomó $\varepsilon = \pi/400$ como radio del entorno local. Por su parte, en búsqueda local iterada se realizaron 20 rearranques con 10 iteraciones de búsqueda local.

Finalmente en el caso de recocidos simulados se realizaron 50 iteraciones. Para el resultado expuesto, se tomó una temperatura inicial de 4000. Cada cinco iteraciones, se redujo la misma en un 1 %. Se realizó una segunda corrida con 100 iteraciones, comenzando con el punto inicial mencionado antes, y reduciendo la temperatura cada 10 iteraciones. No se incluyeron los resultados porque la solución no se modificó en ningún momento a lo largo del algoritmo. Con el objeto de calibrar el método, también optó por tomarse una temperatura de 10000 y de 400, y distintos valores para el entorno de búsqueda local, pero la performance fue incluso peor, siendo la energía final siempre superior a 1000.

En la tabla 3, pueden observarse los resultados obtenidos, mientras que en los gráficos 11 y 12 se encuentran la evolución del mínimo a lo largo de las iteraciones, y la configuración final de la magnetización, respectivamente.

Como puede observarse, los algoritmos genéticos, los cuales comenzaron con poblaciones totalmente al azar, presentaron soluciones que fueron mejorando bastante a lo largo del tiempo. Más aún, el la variante con reproducción por combinación convexa y mutación normal fue el mejor algoritmo entre todos los presentados. Sin embargo, fue el que más tiempo insumió. En general la solución decreció con una pendiente muy marcada en las primeras iteraciones para luego ralentizar su descenso en forma casi asintótica. Esto indica que la cantidad de iteraciones no es tan influyente como la calidad de la población inicial. Consecuencia de estos resultados, podría decirse que los de combinación convexa podrían ser una buena opción si no se tiene información a priori de cómo es la función a minimizar.

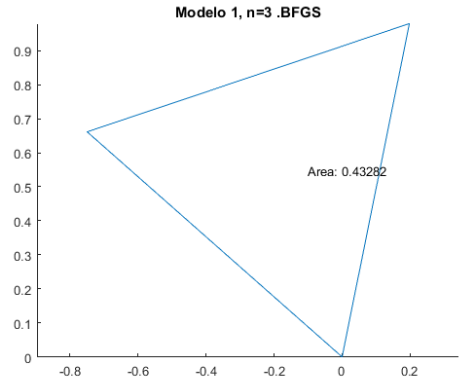
Con respecto a los algoritmos de búsqueda aleatoria, los tres insumieron poco tiempo. El caso de descenso aleatorio puramente global dio una solución que resultó ser la segunda peor de todas. Dada la dimensión del espacio con la que estamos trabajando, este resultado era bastante esperable. Por otro lado, el de

Algoritmo	Mínimo	Tiempo(seg)
Descenso aleatorio (global)	301.11	301
Descenso aleatorio (local)	49.72	389
Descenso aleatorio(local iterada)	50.17	73
Algoritmo genético- ADN, azar	251.64	4735
Algoritmo genético- ADN, Normal	274.2	4897
Algoritmo genético- CombConv, azar	116.051	4783
Algoritmo genético- CombConv, Normal	44.65	5422
Recocido simulado	731.40	3415

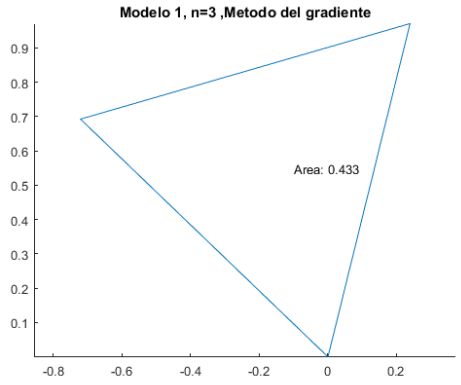
Cuadro 3: Comparación de resultados

búsqueda local, al iniciarse con una solución razonable, mejoró levemente la misma, dando un buen resultado. Por otro lado, el algoritmo de búsqueda local iterada, con el mismo punto inicial, logró mejorar levemente al mismo. Estos métodos tienen la ventaja que requieren mucho menos tiempo que los genéticos, pero en este caso el algoritmo de búsqueda local sólo dio un buen resultado porque se lo inició con un parámetro razonable, algo que algunas veces no puede realizarse. Se realizó una corrida con un punto inicial al azar y en este caso la solución fue equivalente a la obtenida mediante descenso global.

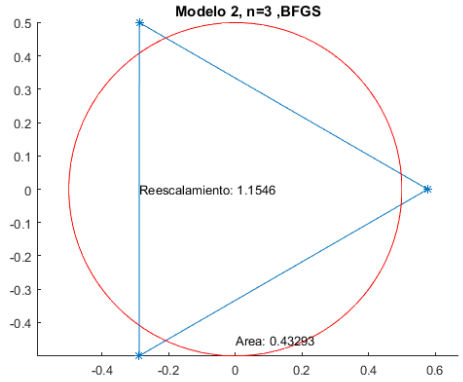
Finalmente el método de recocido simulado fue bastante pobre. Una explicación posible es que la velocidad de descenso de la temperatura fue muy abrupta. El motivo de esto fue que al insumir tanto tiempo, se optó por realizar pocas iteraciones. Si no se hubiera descendido tan abruptamente hubiera terminado asemejándose a un algoritmo de descenso aleatorio.



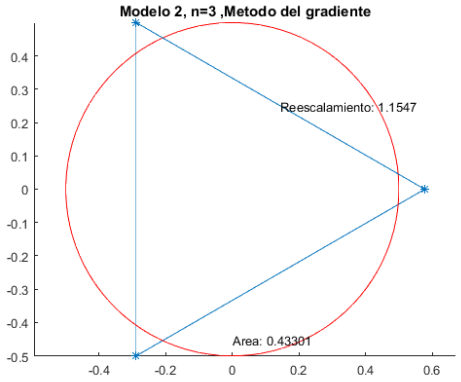
(a)



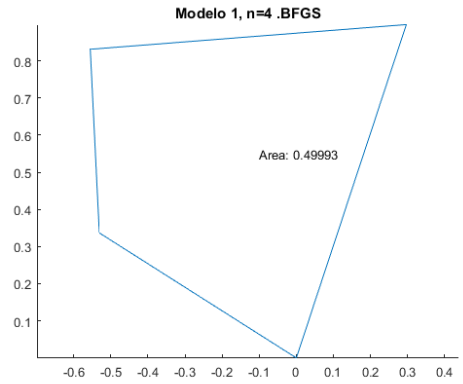
(b)



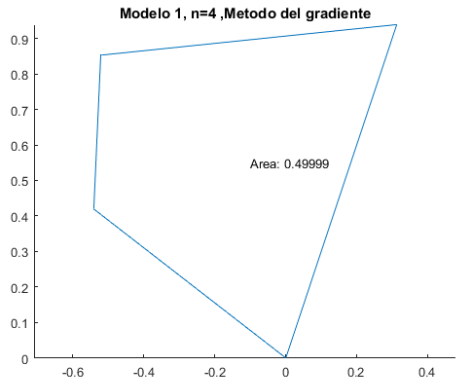
(c)



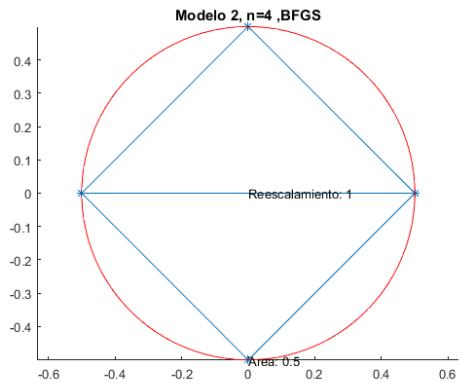
(d)



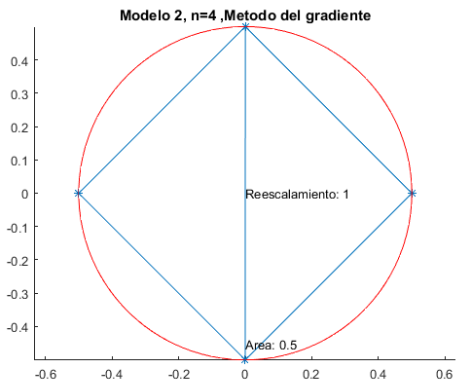
(e)



(f)

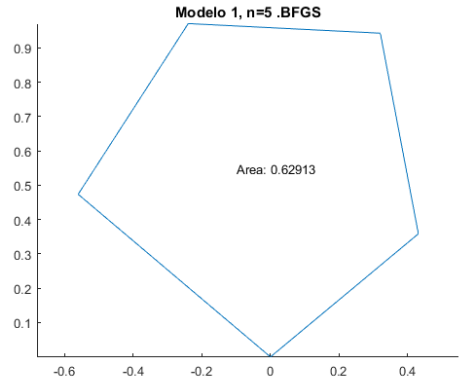


(g)

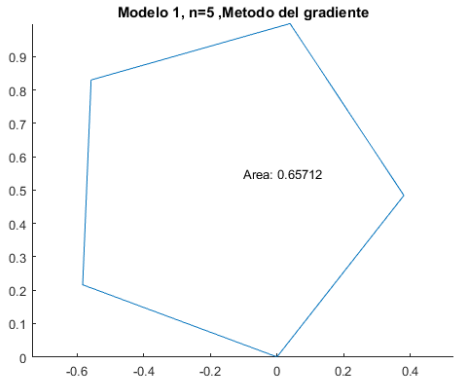


(h)

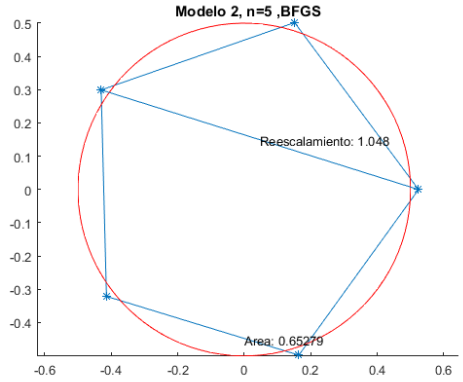
Figura 1: Resultados obtenidos para n=3 y n=4



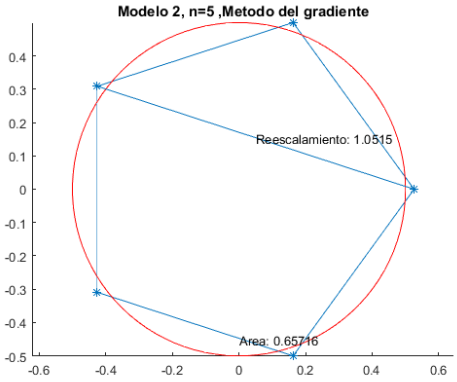
(a)



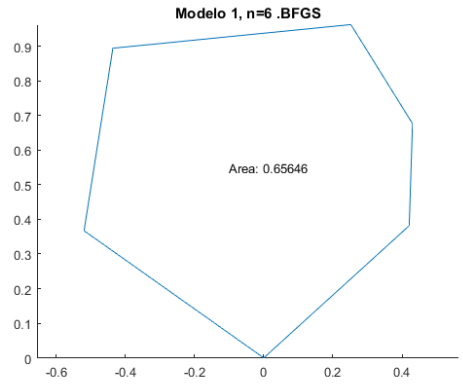
(b)



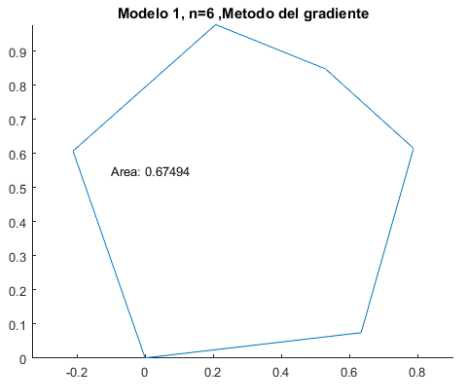
(c)



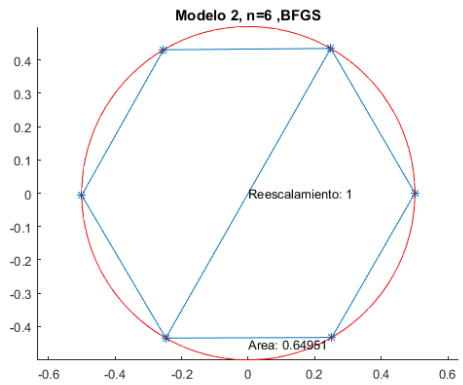
(d)



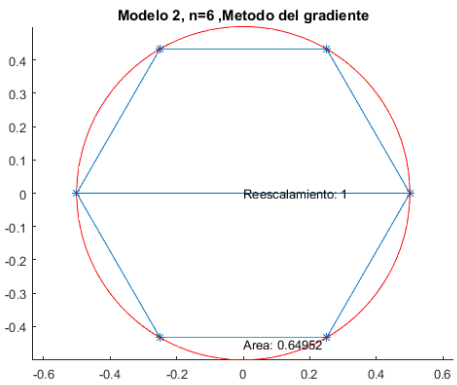
(e)



(f)

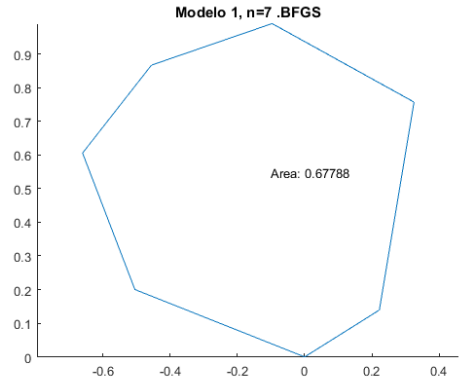


(g)

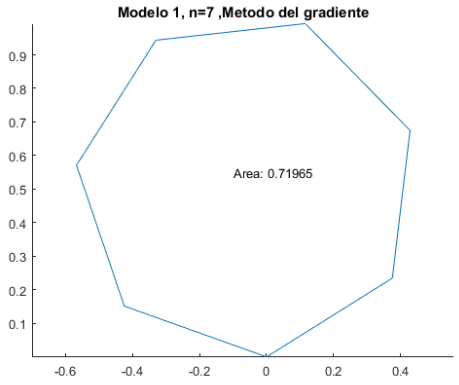


(h)

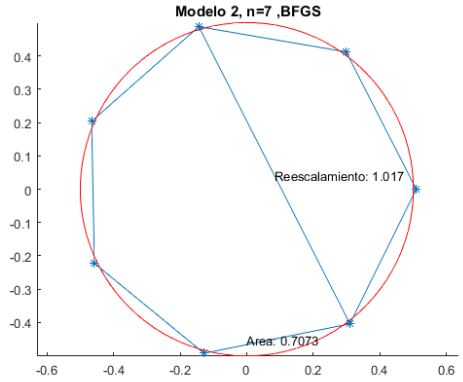
Figura 2: Resultados obtenidos para n=5 y n=6



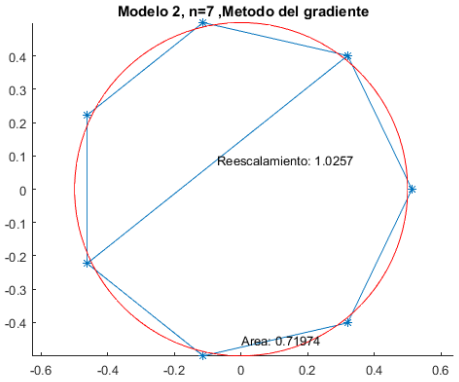
(a)



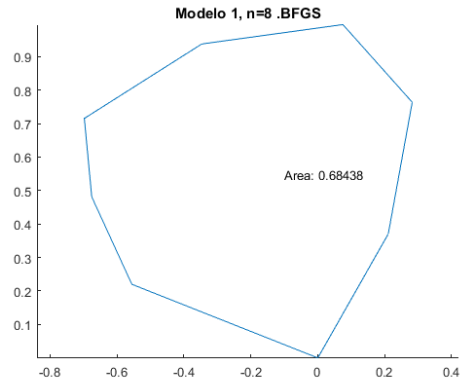
(b)



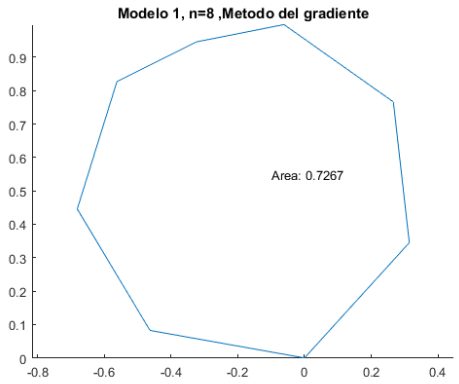
(c)



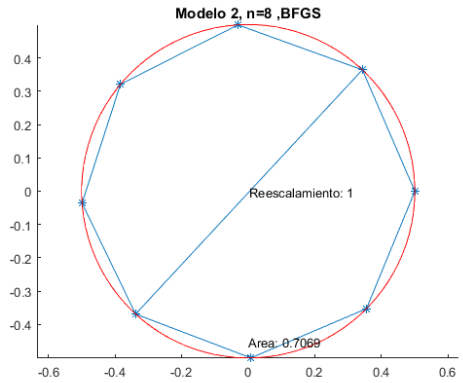
(d)



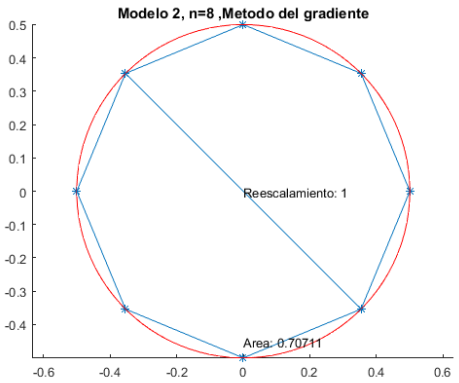
(e)



(f)



(g)



(h)

Figura 3: Resultados obtenidos para n=7 y n=8

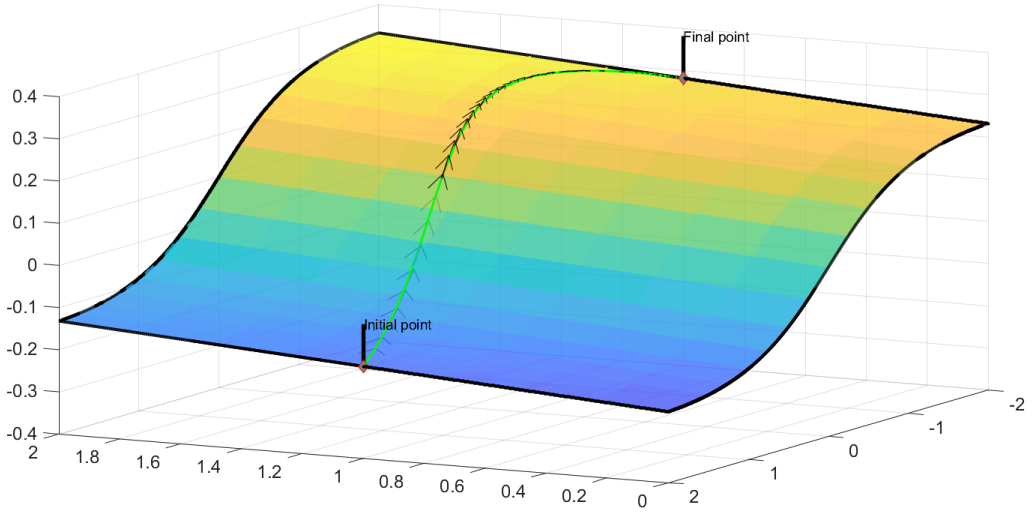


Figura 4: Problema 1 - Trayectoria optimizada

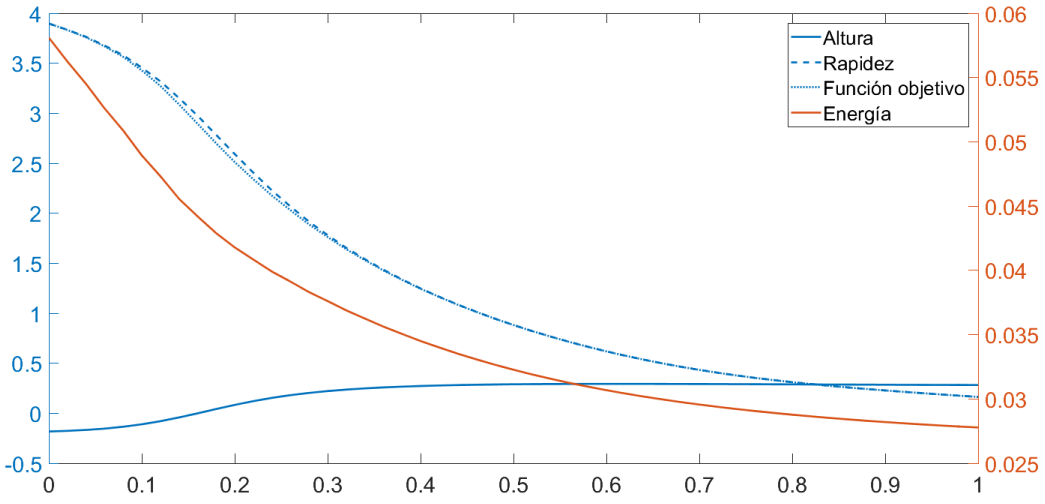


Figura 5: Problema 1 - Evolución de las variables

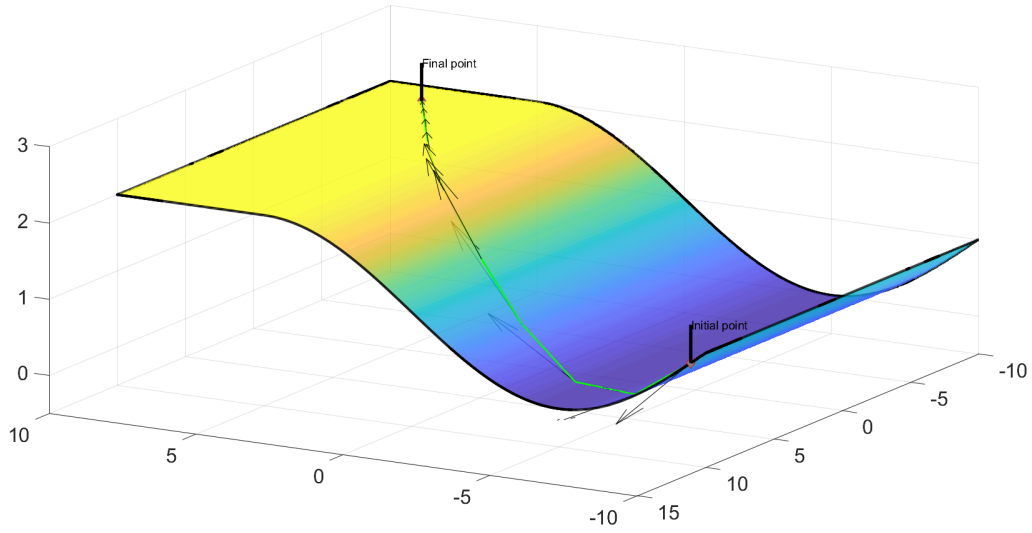


Figura 6: Problema 2 - Trayectoria optimizada

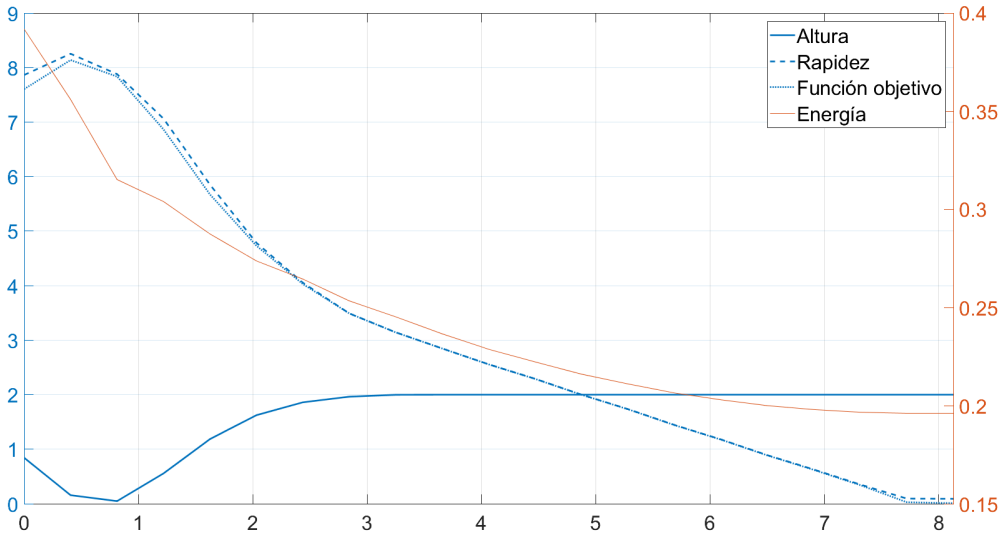


Figura 7: Problema 2 - Evolución de las variables

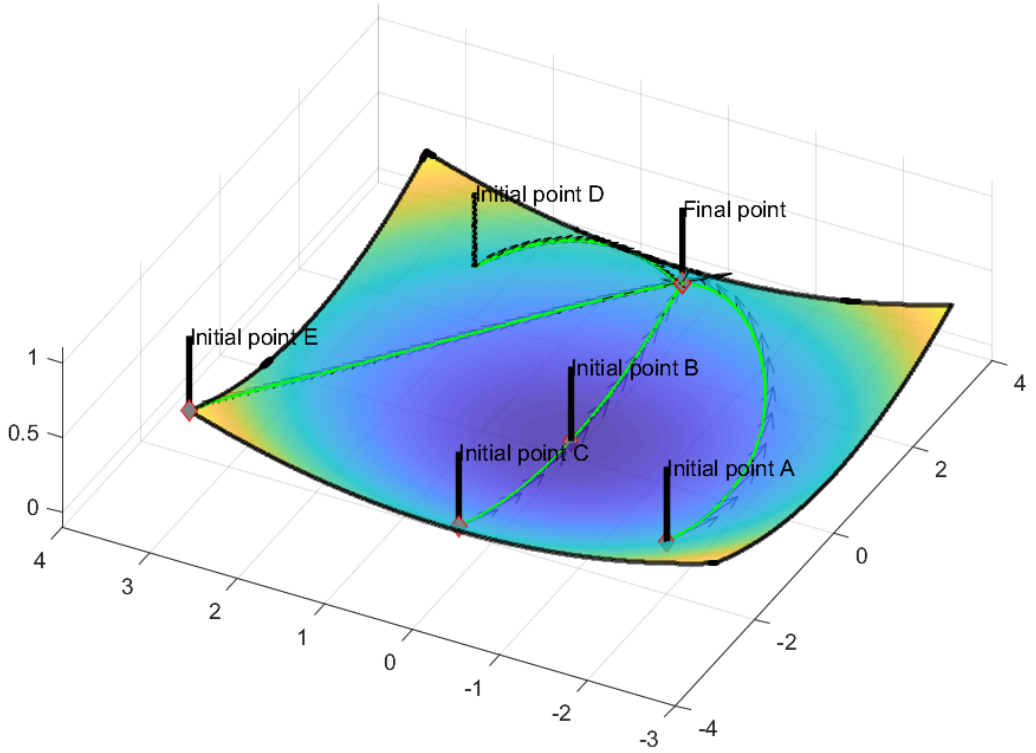


Figura 8: Problema 3 - Trayectorias optimizadas para distintos puntos iniciales

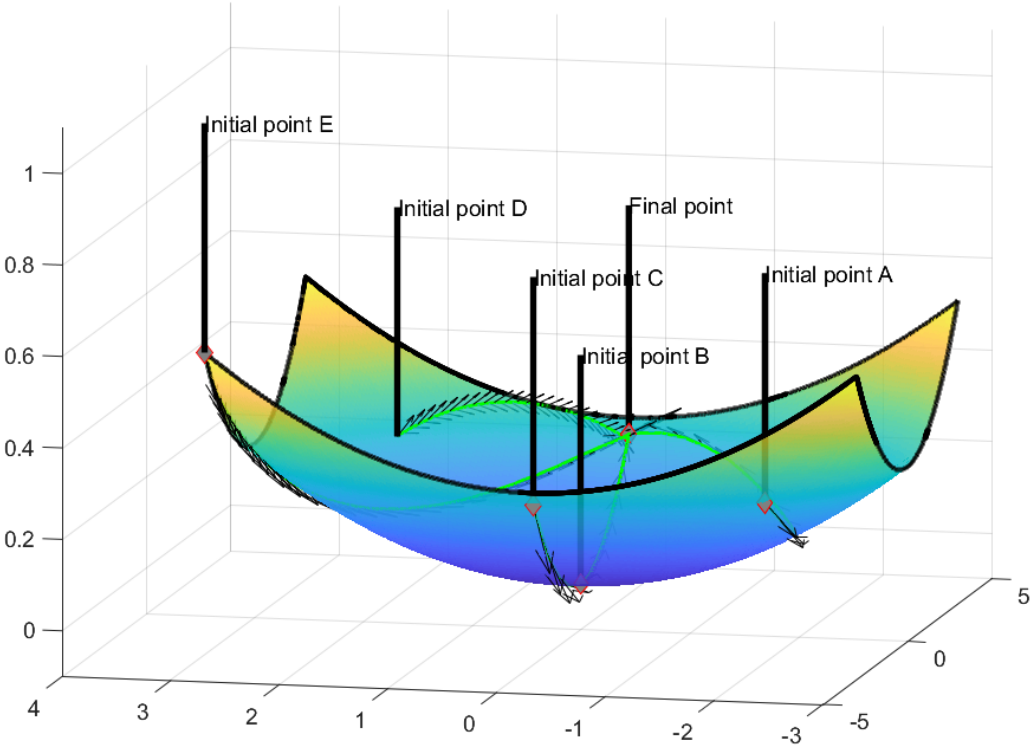


Figura 9: Problema 3 - Trayectorias optimizadas para distintos puntos iniciales

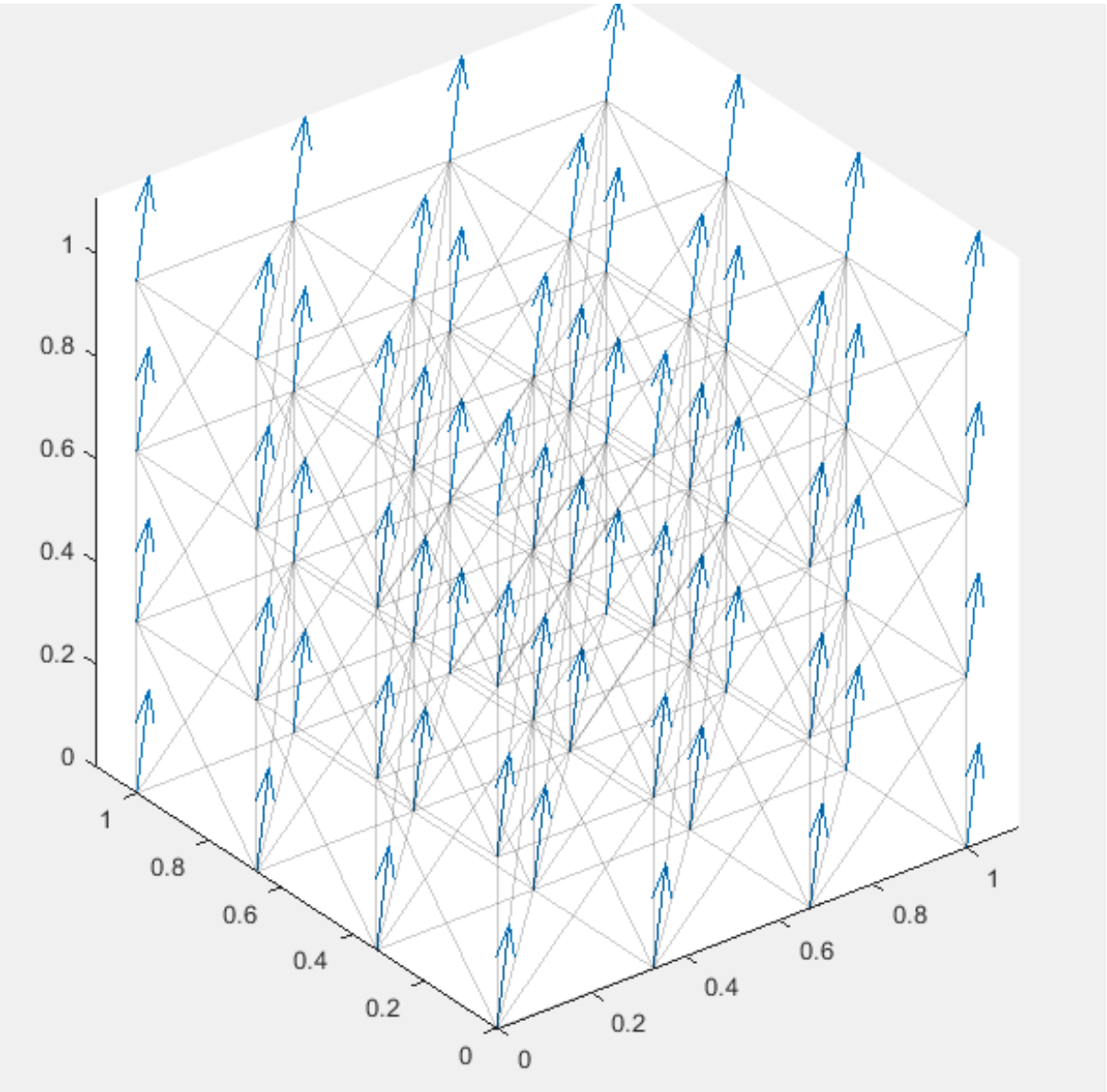
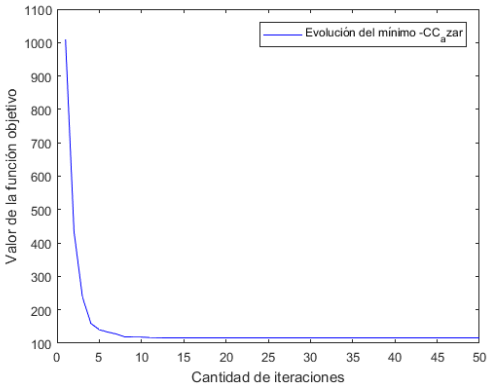
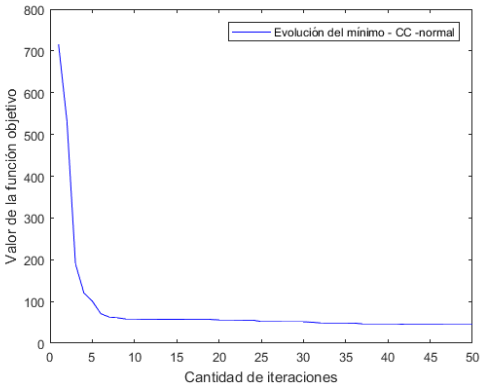


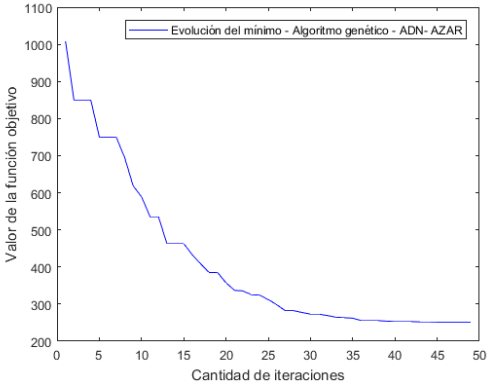
Figura 10: Gráfico de la magnetización en el punto inicial



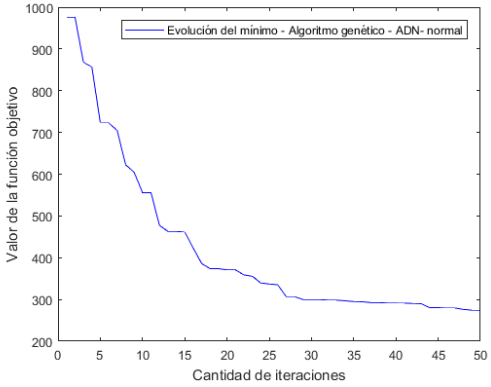
(a) Genético - CC - azar



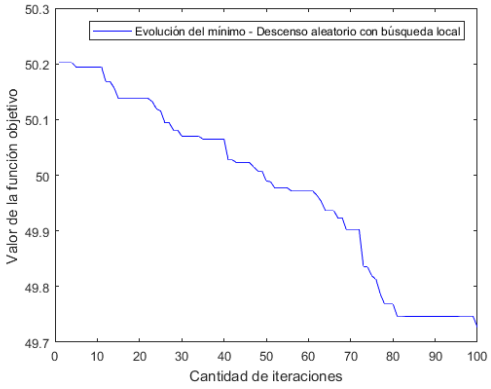
(b) Genético - CC - normal



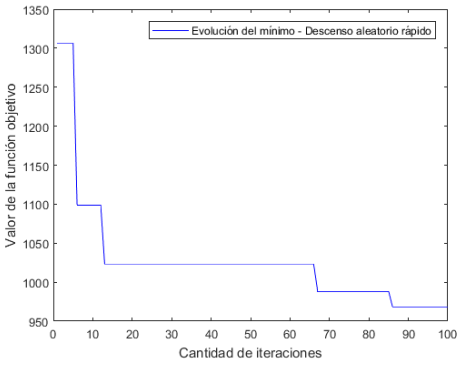
(c) Genético - ADN - azar



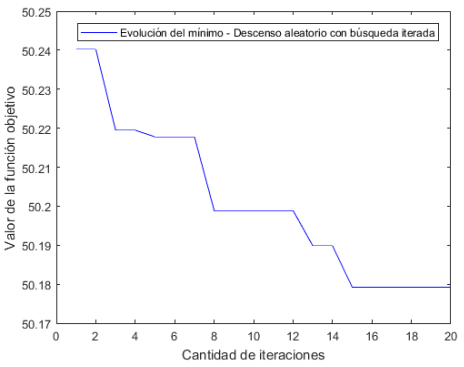
(d) Genético - ADN - normal



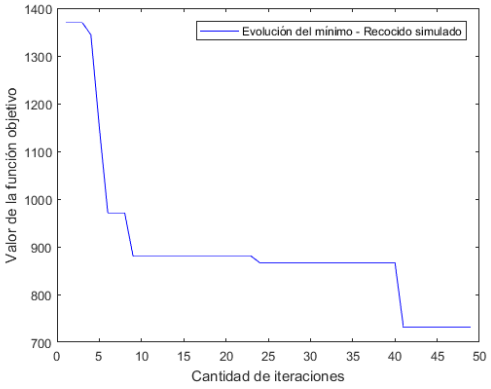
(e) Descenso aleatorio - Búsqueda local



(f) Descenso aleatorio - Búsqueda global

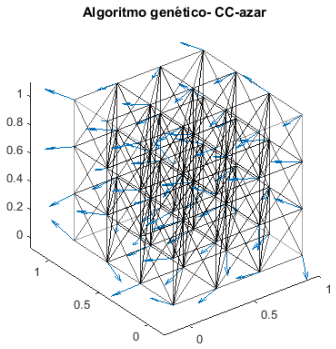


(g) Descenso aleatorio - Búsqueda local iterada

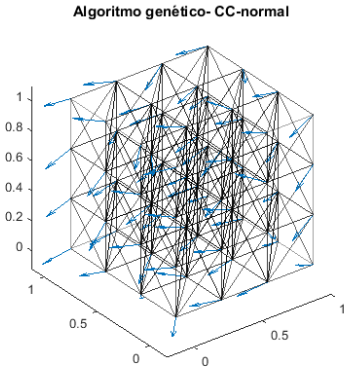


(h) Recocido Simulado

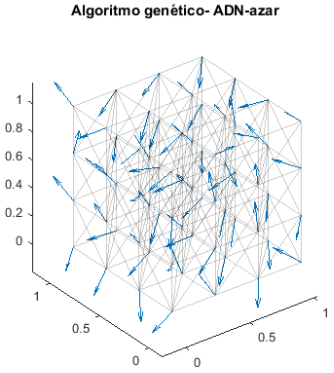
Figura 11: Evolución del mínimo



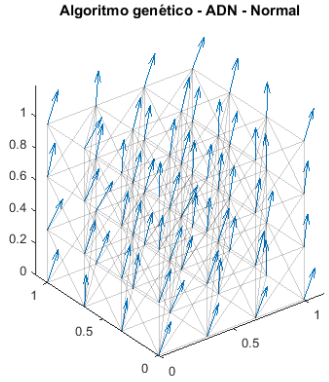
(a) Genético - CC - azar



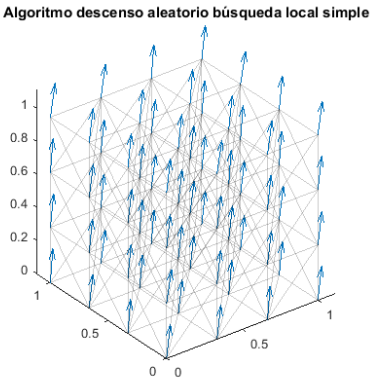
(b) Genético - CC - normal



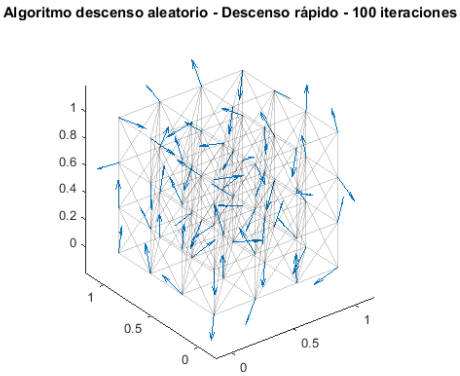
(c) Genético - ADN - azar



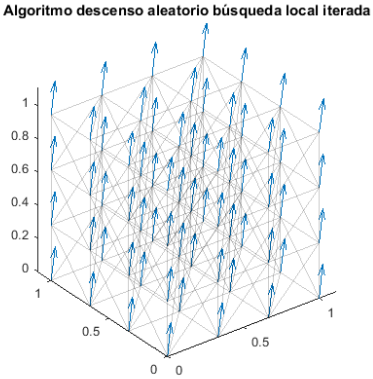
(d) Genético - ADN - normal



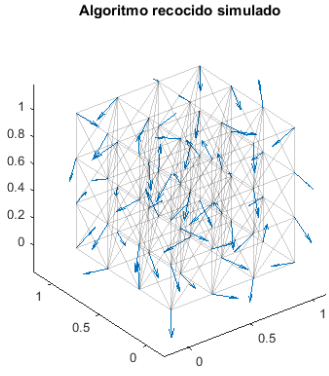
(e) Descenso aleatorio - Búsqueda local



(f) Descenso aleatorio - Búsqueda global



(g) Descenso aleatorio - Búsqueda local iterada



(h) Recocido Simulado

Figura 12: Configuración de la magnetización